

# Lekab Sms Rest Web Service

Lekab Communication Systems AB  
Version 5.0.92, 2021-04-05

# Lekab Sms Rest Web Service

Introduction .....	1
Different authentication methods available for requests .....	1
1. The <code>/send</code> endpoint .....	3
1.1. GET request example .....	3
1.2. POST request example .....	3
1.2.1. Explanation of parameters for <code>/send</code> .....	3
1.2.2. HTTP response to <code>/send</code> .....	4
1.2.3. Explanation of response to <code>/send</code> .....	5
1.2.4. Example Python 3 code .....	5
1.2.5. Tag filter usage .....	6
2. The <code>/send/single</code> endpoint .....	8
2.1. POST request example .....	8
2.1.1. Explanation of parameters for <code>/send/single</code> .....	8
2.1.2. HTTP response to <code>/send/single</code> .....	9
2.1.3. Explanation of response to <code>/send/single</code> .....	9
3. The <code>/status</code> endpoint .....	10
3.1. GET request example .....	10
3.2. POST request example .....	10
3.2.1. Explanation of parameters for <code>/status</code> .....	10
3.2.2. HTTP response to <code>/status</code> .....	11
3.2.3. Explanation of response to <code>/status</code> .....	12
3.2.4. Message status codes .....	12
3.2.5. Non-final statuses .....	12
3.2.6. Successful final status .....	13
3.2.7. Failing final statuses .....	13
3.2.8. Unclear .....	13
3.2.9. Example Python 3 code .....	13
4. The <code>/status/single</code> endpoint .....	14
4.1. POST request example .....	14
4.1.1. Explanation of parameters for <code>/status/single</code> .....	14
4.1.2. HTTP response to <code>/status/single</code> .....	15
4.1.3. Explanation of response to <code>/status/single</code> .....	15
5. The <code>/incoming</code> endpoint .....	16
5.1. GET request example .....	16
5.2. POST request example .....	16
5.2.1. Explanation of parameters for <code>/incoming</code> .....	16
5.2.2. HTTP response to <code>/incoming</code> .....	17
5.2.3. Explanation of response to <code>/incoming</code> .....	18

5.2.4. Example Python 3 code .....	18
6. The <code>/incoming/single</code> endpoint .....	19
6.1. POST request example .....	19
6.1.1. Explanation of parameters for <code>/incoming/single</code> .....	19
6.1.2. HTTP response to <code>/incoming/single</code> .....	20
6.1.3. Explanation of response to <code>/incoming/single</code> .....	20

# Introduction

© 2020 Lekab Communication Systems AB. Version 5.0.92, 2021-04-05.

This is a Web Service using `HTTP GET` or `HTTP POST` requests to send SMS, read status of SMS and retrieve incoming SMS.

There are three main endpoints: `/send`, `/status` and `/incoming`, with the obvious purposes.

Each of the endpoints supports the same function with `GET` and `POST`, but in the `GET` case, parameters are given in the calling URL (after a `?` sign, separated by `&` signs) , while in the `POST` case the parameters are given in a json document in the `HTTP POST` request body. `UTF-8` encoding is assumed in all `HTTP` bodies. The `GET` case, for historical reasons, allows either `ISO-8859-1` or `UTF-8` based url-encoding of the message using different query parameters.

Both the `GET` and the `POST` return responses in the `HTTP` response body as a json document.

Each of the endpoints also has a variant, single data object `HTTP POST` endpoint (send one SMS, retrieve one status or one incoming SMS), aimed for integration into less sophisticated calling applications which cannot handle lists of objects: `/send/single`, `/status/single` and `/incoming/single`.

The format of the input and output json documents and the input url parameters are described below.

## Different authentication methods available for requests

Every request to the service must include authentication, i.e. username and password (or equivalent). For `POST` requests these can be given in in the corresponding fields in the `JSON` document which is sent in the (automatically `HTTPS = SSL/TLS` encrypted) `HTTP` body. Since `GET` requests cannot have a body, instead username and password can be sent in the `U` and `P` url parameters. Note that everything in the URL after the host name is also part of the encrypted request, so url parameters are as safe during transfer as parameters in the body.

We also offer three different alternative ways of supplying these username and password credentials available in both the `GET` and `POST` cases:

1. Username and password can be given as standard Basic authentication, in which the header `Authorization` should have the value `Basic + token`, where the token is the `Base64` encoding of (a `UTF-8` byte array representation of) `username:password`. Here `testuser:testpass` will be encoded as `dGVzdHVzZXI6dGVzdHBhc3M=`
2. Username and password can be given in the `HTTP` headers, `X-Lekab-Userid` and `X-Lekab-Password`, respectively. The values have to be the `Base64` encoding of (a `UTF-8` byte array representation of) the username or password to allow non-US-ASCII characters. Here `testuser` will be encoded as `dGVzdHVzZXI=` and `testpass` as `dGVzdHBhc3M=`
3. An API key obtained from Lekab can be used as a query parameter `key`, as the value of the header `X-API-Key` or as the field `apikey` in the `POST` request body. The length

of the key varies depending on the length of the username (which is contained within the key). `TUxVomRHVnpkSFZ6WlhJOj1KUUczTXU2TVZVU1Exd3Y` is a possible example key for the username `testuser`. The key is independent of the account password.

If any of the alternative methods of authentication are used, parameter values pertaining to other authentication methods should be omitted or set to the empty string `""`.

# Chapter 1. The /send endpoint

Used for sending SMS

## 1.1. GET request example

e.g. from web browser or curl

```
curl
https://secure.lekab.com/restsms/lekabrest/send?U=testuser&P=testpass&T=46701234567,46CALLMENOW&F=LEKAB&M8=Hall%C3%A5+d%C3%A4r!&2=TRUE&X=CONV123
```

## 1.2. POST request example

probably from an application

```
https://secure.lekab.com/restsms/lekabrest/send
```

With the contents of the HTTP body:

```
{ "username": "testuser", "password": "testpass", "from": "LEKAB",
  "to": [ "46701234567", "46CALLMENOW" ],
  "message": "Hallå där!", "twoway": true, "conversation": "CONV123" }
```

### 1.2.1. Explanation of parameters for /send

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
username	U	string	string	username of the LEKAB SMS sending account
password	P	string	string	password of the LEKAB SMS sending account
apikey	key	string	string	API key of the LEKAB SMS sending account
to	T	json list of string	comma separated strings	recipient phone number list
from	F	string	string	sender id of the SMS (ignored for two-way SMS)
message	M	string UTF-8	string URL-encoded ISO-8859-1	Message to send (alternative GET query param, only specify one)
message	M8	string UTF-8	string URL-encoded UTF-8	Message to send (alternative GET query param, only specify one)
twoway	2	boolean (default false)	T, TRUE, Y or YES	sender id from reply number pool

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
conversation	X	string	string	conversation id not sent but echoed with two-way reply
costcenter	C	string	string	grouping in billing
flash	Z	boolean (default false)	T, TRUE, Y or YES	send flash SMS
validminutes	V	string containing integer	string	validity time of SMS (default is 1440 i.e. 24h)
shownumberparts	N	boolean (default false)	T, TRUE, Y or YES	number of SMS parts of multipart message in HTTP response
defaultcountrycode	D	string containing digits	string containing digits	replaces leading zero in <b>to</b> numbers starting with single zero
toaddressbookcustomtagfilter	A	string containing tag filter	string URL-encoded UTF-8 containing tag filter	filter specifying combination of tags in the user's address book
toaddressbooktagfilterjson	(n/a)	json list of filter parts (see below)	(n/a)	filter specifying combination of tags in the user's address book
toaddressbooksavedtagfilters	S	json list of saved tag filter names	comma separated strings each URL-encoded UTF-8 containing saved filter name	names of saved tag filters from the user's address book
toaddressbookgroups	G	json list of group/distribution list names	comma separated strings each URL-encoded UTF-8 containing group/distribution list name	names of groups/distribution lists from the user's address book
replacecharacters	R	boolean (default false)	T, TRUE, Y or YES	Replace certain characters not in the GSM-7 standard character set with similar characters in that set (e.g. non-breaking space u00A0 replaced by ordinary space u0020). Only applied if enabled for the account; contact Lekab sales for details.
scheduletime	AT	string containing time	string URL-encoded UTF-8 containing time	Schedule sendout for sending in the future. ISO 8601 standard format "2022-02-14T09:00:00Z" or "2022-02-14T11:00:00+0200"

### 1.2.2. HTTP response to /send

A successful request will return 200 OK and a json document of either of the two following formats (depending on the shownumberparts/N parameter). Note that the request for sending is successful even if the send status of some SMS sendings are failed. A successfully sent message may sometimes not be delivered due to external factors (phone turned off, phone subscription expired, no such subscriber), but such requests will be reported as successful by this endpoint, and the later fortune of the message can be followed via the /status or /status/single endpoints. A scheduled message is reported as accepted, and will have status SCHEDULED until the scheduled send time. The Content-Type header of the response is application/json for all responses.

```

{
  "accepted" : [ {
    "to" : "46701234567",
    "id" : "354284289"
  } ],
  "rejected" : [ "46CALLMENOW" ]
}

```

Or with number of parts enabled

```

{
  "accepted" : [ {
    "to" : "46701234567",
    "id" : "354284289",
    "parts" : "1"
  } ],
  "rejected" : [ "46CALLMENOW" ]
}

```

Note that in this example, the mobile recipient number was accepted while the alphanumeric recipient number was rejected.

### 1.2.3. Explanation of response to /send

POST json key	json value (strings quoted)	
accepted	json list of json documents	list of acknowledged queued SMS, never empty because no queued SMS will cause a HTTP 400 status
rejected	json list of string	list of rejected recipients
to	string	recipient phone number
id	string	id of the SMS (use this for future references to this SMS, e.g. for status queries)
parts	string	number of SMS parts (decimal digits only) sent, since a SMS has a max length depending on character set

### 1.2.4. Example Python 3 code



```

import json
import requests

sendreq = {"username" : "testuser", "password": "testpass", "from":
"Lekab", "to":["46700112233","46CALLMENOW"], "message":"Tjo flöjt!"}
sendreq_json = json.dumps(sendreq)
url = 'https://secure.lekab.com/restsms/lekabrest/send'
response = requests.post(url, data=sendreq_json)
sendresp = response.json()
for a in sendresp["accepted"]:
    print("SMS to recipient " + a["to"] + " got message id " + a["id"])
for r in sendresp["rejected"]:
    print("Cannot send to " + r)

```

will output

```

SMS to recipient 46700112233 got message id 6205
Cannot send to 46CALLMENOW

```

### 1.2.5. Tag filter usage

A user, or a company with many users, can have an address book in the Lekab SMS system, with contacts that each contain a phonenumber that can receive SMS messages. These contacts can be marked with `tags` consisting of a tag type and a tag value. For instance, a contact can be marked with the tag `Base:STO` if the person is based in Stockholm. The same contact can have many tags, for instance `Group:Management` or `On call:Yes`. Which tags types and tag values are used is up to the user/company, but they should preferably consist of letters and numbers, and can especially not contain the characters `;`, `|` or `:`. A tag filter in the SMS rest interface consists of a list of filter parts, which are connected by logical AND (a contact is selected only if it fulfills every part of the filter). A filter part consists of a tag type and a list of tag values. A contact fulfills the filter part criterion if it has a tag with the given tag type and one of the given tag values (logical OR). Filters can be given either as a string in GET or POST or as a json document in the POST input. Using the string notation, the example contact would fulfill a `Base:STO|OSL|CPH;On call:Yes` filter, which consists of two filter parts, one saying that the contact needs to have a `Base:STO` or a `Base:OSL` or a `Base:CPH` tag, and the other that the contact needs to have an `On call:Yes` tag. Note that filter parts are separated by a semicolon, tag type and tag values are separated by a colon and tag values are separated by vertical bars (UNIX pipe symbols).

A user may have access to several address books, but only the address book set as default address book for the user will be searched with the tag filter.

If a tag filter is given, the `to` parameter is optional, and any numbers selected both by the filter and by the `to` parameter will not receive duplicate messages but only one. Only one tag filter can be given per send call.

In the following three parameter examples (two POST, one GET), the same tag filter is specified. Note the URL-encoding of the filter string in the GET parameter.

```
{ "username": "testuser", "password": "testpass", "from": "LEKAB",  
  "toaddressbookcustomtagfilter": "Base:STO|OSL|CPH;On call:Yes",  
  "message": "Hallå där!", "conversation": "CONV123" }
```

```
{ "username": "testuser", "password": "testpass", "from": "LEKAB",  
  "toaddressbooktagfilterjson": [ { "tagtype": "Base", "tagvalues": [ "STO",  
    "OSL", "CPH" ] }, { "tagtype": "On call", "tagvalues": [ "Yes" ] } ],  
  "message": "Hallå där!", "conversation": "CONV123" }
```

```
/send?U=testuser&P=testpass&A=Base%3aSTO%7cOSL%7cCPH%3bOn%20call%3aYes&  
F=LEKAB&M8=Hall%C3%A5+d%C3%A4r!&X=CONV123
```

Tag filters may be saved in the address book for repeated use, under a name. Note the `toaddressbooksavedtagfilters/S` parameter which takes a list of such names.

```
{ "username": "testuser", "password": "testpass", "from": "LEKAB",  
  "toaddressbooksavedtagfilters": [ "Pilots Helsinki", "Security Helsinki  
" ],  
  "message": "Hallå där!", "conversation": "CONV123" }
```

```
/send?U=testuser&P=testpass&S=Pilots%20Helsinki,Security%20Helsinki&F=L  
EKAB&M8=Hall%C3%A5+d%C3%A4r!&X=CONV123
```

Lekab's recommendation is to use the more flexible tag filters, but legacy customers using named groups/distribution lists can give a list of such names in the `toaddressbookgroups/G` parameter.

```
{ "username": "testuser", "password": "testpass", "from": "LEKAB",  
  "toaddressbookgroups": [ "IT Dept", "Managers" ],  
  "message": "Hallå där!", "conversation": "CONV123" }
```

```
/send?U=testuser&P=testpass&G=IT%20Dept,Managers&F=LEKAB&M8=Hall%C3%A5+  
d%C3%A4r!&X=CONV123
```

## Chapter 2. The /send/single endpoint

Used for sending a single SMS

### 2.1. POST request example

probably from an application

```
https://secure.lekab.com/restsms/lekabrest/send/single
```

With the contents of the HTTP body:

```
{ "username": "testuser", "password": "testpass", "from": "LEKAB",  
  "to": "46701234567",  
  "message": "Hallå där!", "twoway": true, "conversation": "CONV123" }
```

#### 2.1.1. Explanation of parameters for /send/single

The /send/single endpoint accepts a single string field as the "to" parameter, and does not accept any parameters that may lead to sending messages to more than one recipient (groups, tag filters etc.)

POST json key	json value (strings quoted)	
username	string	username of the LEKAB SMS sending account
password	string	password of the LEKAB SMS sending account
apikey	string	API key of the LEKAB SMS sending account
to	string	recipient phone number
from	string	sender id of the SMS (ignored for two-way SMS)
message	string UTF-8	Message to send
twoway	boolean (default false)	use sender id from reply number pool
conversation	string	conversation id not sent but echoed with two-way reply
costcenter	string	grouping in billing
flash	boolean (default false)	send flash SMS
validminutes	string containing integer	validity time of SMS (default is 1440 i.e. 24h)
defaultcountrycode	string containing digits	replaces leading zero in <b>to</b> number starting with single zero

POST json key	json value (strings quoted)	
replacecharacters	boolean (default false)	Replace certain characters not in the GSM-7 standard character set with similar characters in that set (e.g. non-breaking space u00A0 replaced by ordinary space u0020). Only applied if enabled for the account; contact Lekab sales for details.
scheduledtime	AT	string containing time

### 2.1.2. HTTP response to /send/single

A successful request will return 200 OK and a json document of the following format. A failure to attempt sending due e.g. to an invalid recipient phone number will return an appropriate error code, e.g. 400 Invalid request. A successfully sent message may sometimes not be delivered due to external factors (phone turned off, phone subscription expired, no such subscriber), but such requests will be reported as successful by this endpoint, and the later fortune of the message can be followed via the `/status` or `/status/single` endpoints. A scheduled message is reported as accepted, and will have status `SCHEDULED` until the scheduled send time. The `Content-Type` header of the response is `application/json` for all responses.

```
{
  "to" : "46701234567",
  "id" : "354284289",
  "parts" : "1"
}
```

### 2.1.3. Explanation of response to /send/single

POST json key	json value (strings quoted)	
to	string	recipient phone number
id	string	id of the SMS (use this for future references to this SMS, e.g. for status queries)
parts	string	number of SMS parts (decimal digits only) sent, since SMS messages are sent in parts, each with a max length depending on character set

## Chapter 3. The `/status` endpoint

Used to retrieve the status of SMS that were sent earlier.

Reading statuses will by default mark them as read, unless the `markasread` parameter is explicitly set to `false`. When the status of a message changes, the status is set to unread by the system

This endpoint can either retrieve unread statuses of SMS messages sent, or retrieve the statuses for a list of given SMS message ids (independent on whether they were read before or not).

Reading the status of a message will by default mark the status as read, but this can be avoided by setting the `markasread` parameter to `FALSE`. This default value is `TRUE` both when reading unread statuses and when reading listed statuses. If `markasread` is set to `FALSE`, the same statuses will be retrieved again on the next call.

### 3.1. GET request example

e.g. from web browser or curl

```
curl
https://secure.lekab.com/restsms/lekabrest/status?U=testuser&P=testpass
&R=FALSE&I=1088,4140,4118,4243,4412
```

### 3.2. POST request example

Probably from an application

```
https://secure.lekab.com/restsms/lekabrest/status
```

with the contents of the HTTP body:

```
{
  "username" : "testuser",
  "password" : "testpass",
  "markasread" : false,
  "id" : [ "1088", "4140", "4118", "4243", "4412" ]
}
```

#### 3.2.1. Explanation of parameters for `/status`

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
username	U	string	string	username of the LEKAB SMS sending account
password	P	string	string	password of the LEKAB SMS sending account

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
apikey	key	string	string	API key of the LEKAB SMS sending account
id	l	json list of string	comma separated strings	list of ids of the SMS for which send status is to be retrieved
maxnum	N	integer (default 100)	integer	max number of statuses to retrieve (ignored if ids are listed)
markasread	R	boolean (default true)	F, FALSE, N or NO	flag whether the status should be marked as read (defaults to true)

A `POST` call with default parameters, where the authorization credentials are supplied via Basic Authentication or X-Lekab-headers can supply either a zero-length body or an empty `JSON` document with the same result.

### 3.2.2. HTTP response to /status

A successful request will return 200 OK and a json document of the following format. The `Content-Type` header of the response is `application/json` for all responses.

```
{
  "statuses" : [ {
    "to" : "46700123456",
    "from" : "46737494333249",
    "id" : "1088",
    "status" : "DELIVERED",
    "statuscode" : "2",
    "conversation" : "",
    "time" : "1467132305000"
  }, {
    "to" : "46705123456",
    "from" : "Lekab",
    "id" : "4243",
    "status" : "QUEUED",
    "statuscode" : "0",
    "conversation" : "",
    "time" : "1476454236000"
  }, {
    "to" : "46702345678",
    "from" : "46737494333295",
    "id" : "4412",
    "status" : "UNDELIVERABLE",
    "statuscode" : "6",
    "conversation" : "74867653486858240",
    "time" : "1477311457000"
  } ],
  "notfound" : [ "4140", "4118" ]
}
```

Note that in this example, three of the listed statuses were found, while two were not. If the user is not allowed to read a status for a certain id, because that message belongs to another user, this is treated as not found.

### 3.2.3. Explanation of response to /status

POST json key	json value (strings quoted)	
statuses	json list of json documents	list of send statuses retrieved
notfound	json list of string	list of message ids for which no status could be retrieved
to	string	recipient phone number
from	string	sender id (number or alphanumeric)
id	string	id of the SMS
status	string	name of the status state of the message (DELIVERED is good)
statuscode	string containing integer 0 to 15	integer code of the status state of the message
conversation	string	conversation id given when sending the message
time	string containing long integer	timestamp in milliseconds after the epoch (1970-01-01 00:00:00 Z)

### 3.2.4. Message status codes

The following message status codes can be received in the message status.

stat usc ode	status	Description
0	QUEUED	Queued for delivery
1	SENT	Sent to operator
2	DELIVERED	Delivered to the mobile station
3	DELETED	The message was deleted
4	EXPIRED	The message has expired
5	REJECTED	The message was rejected by the operator
6	UNDELIVERABLE	The message could not be delivered
7	ACCEPTED	The message was accepted by the operator
8	ABSENTSUBSCRIBER	The subscribers mobile station is switched off
9	UNKNOWNSUBSCRIBER	The subscriber is not known
10	INVALIDDESTINATION	The destination address is invalid
11	SUBSCRIBERERROR	The mobile station can not receive the message
12	UNKNOWN	The status of the message is unknown
13	ERROR	Internal error when sending the message
14	SCHEDULED	Not yet sent but scheduled for sending
15	CANCELED	Scheduled message was canceled before sending

### 3.2.5. Non-final statuses

A later status update from the mobile carrier is quite likely to change the value.

QUEUED (0), SENT (1), SCHEDULED (14)

When the status is updated, the **read** flag is cleared to unread status, and the new status will be retrieved the next time reading. Note that when the recipient's phone is turned off, this is where the status is stuck until the phone is turned on or the message expires.

### 3.2.6. Successful final status

The recipient's phone has supposedly acknowledged receipt of this message.

DELIVERED (2)

### 3.2.7. Failing final statuses

The recipient will not get this message.

DELETED (3), EXPIRED (4), REJECTED (5), UNDELIVERABLE (6),  
ABSENTSUBSCRIBER (8), UNKNOWNSUBSCRIBER (9), INVALIDDESTINATION (10),  
SUBSCRIBERERROR (11), ERROR (13), CANCELED (15)

### 3.2.8. Unclear

Probably failing final statuses (probably the mobile carrier has lost or dumped the message, but it may suddenly turn up).

ACCEPTED (7), UNKNOWN (12)

### 3.2.9. Example Python 3 code

```
import json
import requests

statuses = {"username" : "testuser", "password": "testpass",
"markasread" : False, "id" : [ "6202", "6203", "6204" ]}
statuses_json = json.dumps(statuses)
url = 'https://secure.lekab.com/restsms/lekabrest/status'
response = requests.post(url, data=statuses_json)
statusresp = response.json()
for s in statusresp["statuses"]:
    print("id=" + s["id"] + ", status=" + s["status"])
```

will output

```
id=6202, status=DELIVERED
id=6203, status=UNDELIVERABLE
id=6204, status=QUEUED
```



## Chapter 4. The `/status/single` endpoint

Used to retrieve the status of one (1) SMS that was sent earlier.

Reading a status will by default mark it as read, unless the `markasread` parameter is explicitly set to `false`. When the status of a message changes, the status is set to unread by the system.

This endpoint can either retrieve an unread status of a sent SMS message, or retrieve the status for a given SMS message id (independent on whether it was read before or not).

Reading the status of a message will by default mark the status as read, but this can be avoided by setting the `markasread` parameter to `false`. This default value is `true` both when reading unread statuses and when reading status by id. If `markasread` is set to `false`, the same status will, in most cases, be retrieved again on the next call (unless the status changed in the mean time).

### 4.1. POST request example

Probably from an application

```
https://secure.lekab.com/restsms/lekabrest/status/single
```

with the contents of the HTTP body:

```
{
  "username" : "testuser",
  "password" : "testpass",
  "markasread" : false,
  "id" : "1088"
}
```

#### 4.1.1. Explanation of parameters for `/status/single`

The `/status/single` endpoint accepts a single string field as the "id" parameter, and if no id is given a maximum of one (1) unread status change is returned.

POST json key	json value (strings quoted)	
username	string	username of the LEKAB SMS sending account
password	string	password of the LEKAB SMS sending account
apikey	string	API key of the LEKAB SMS sending account
id	string	id of the SMS for which send status is to be retrieved
markasread	boolean (default true)	flag whether the status should be marked as read (defaults to true)

A `POST` call with default parameters, where the authorization credentials are supplied via

Basic Authentication or X-Lekab-headers can supply either a zero-length body or an empty JSON document with the same result.

### 4.1.2. HTTP response to /status/single

A successful request will return 200 OK and a json document of the following format. The Content-Type header of the response is application/json for all responses.

```
{
  "to" : "46700123456",
  "from" : "46737494333249",
  "id" : "1088",
  "status" : "DELIVERED",
  "statuscode" : "2",
  "conversation" : "",
  "time" : "1467132305000"
}
```

If the id given does not correspond to a sent SMS that the requesting user is allowed to see, or if no id is given and no unread status change is available, a 404 Not found error will be returned.

### 4.1.3. Explanation of response to /status/single

POST json key	json value (strings quoted)	
to	string	recipient phone number
from	string	sender id (number or alphanumeric)
id	string	id of the SMS
status	string	name of the status state of the message (DELIVERED is good)
statuscode	string containing integer 0 to 15	integer code of the status state of the message
conversation	string	conversation id given when sending the message
time	string containing long integer	timestamp in milliseconds after the epoch (1970-01-01 00:00:00 Z)

The status values are described above, in the explanations of the /status endpoint.

## Chapter 5. The `/incoming` endpoint

Used to retrieve incoming SMS that were either sent to a short or long number rented by the user, or to a number pool number in response to a two-way SMS.

Reading incoming SMS will by default mark them as read, unless the `markasread` parameter is explicitly set to `false`

This endpoint can either retrieve unread incoming SMS messages, or retrieve incoming SMS messages whose message ids are given (independent of whether they were read before or not).

Reading a message will by default mark the message as read, but this can be avoided by setting the `markasread` parameter to `FALSE`. This default value is `TRUE` both when reading unread messages and when reading id listed messages. If `markasread` is set to `FALSE`, the same messages will be retrieved again on the next call.

### 5.1. GET request example

e.g. from web browser or curl

```
curl
https://secure.lekab.com/restsms/lekabrest/incoming?U=testuser&P=testpass&R=FALSE&N=10&G=Y
```

### 5.2. POST request example

Probably from an application

```
https://secure.lekab.com/restsms/lekabrest/incoming
```

With the contents of the HTTP body:

```
{
  "username" : "testuser",
  "password" : "testpass",
  "markasread" : false,
  "maxnum" : 10,
  "getoriginal" : true
}
```

#### 5.2.1. Explanation of parameters for `/incoming`

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
username	U	string	string	username of the LEKAB SMS sending account

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
password	P	string	string	password of the LEKAB SMS sending account
apikey	key	string	string	API key of the LEKAB SMS sending account
id	I	json list of string	comma separated strings	list of ids of the SMS for which send status is to be retrieved
maxnum	N	integer (default 100)	integer	max number of statuses to retrieve (ignored if ids are listed)
markasread	R	boolean (default true)	F, FALSE, N or NO	flag whether the incoming message should be marked as read (defaults to true)
getoriginal	G	boolean (default false)	T, TRUE, Y or YES	should original SMS text in a two-way conversation be retrieved?

A POST call with default parameters, where the authorization credentials are supplied via Basic Authentication or X-Lekab-headers can supply either a zero-length body or an empty JSON document with the same result.

### 5.2.2. HTTP response to /incoming

A successful request will return 200 OK and a json document of the following format. The Content-Type header of the response is application/json for all responses.

```
{
  "incoming" : [ {
    "from" : "46701234567",
    "to" : "54321",
    "id" : "1077",
    "message" : "Please send more info about the club",
    "conversation" : "",
    "resptoid" : "",
    "origmess" : "",
    "time" : "1478538376000"
  }, {
    "from" : "46711223344",
    "to" : "46737494333766",
    "id" : "323",
    "message" : "Yes I would love to",
    "conversation" : "67259314888265728",
    "resptoid" : "3403",
    "origmess" : "Will you join us at the pub after?",
    "time" : "1475497511000"
  } ],
  "notfound" : [ ]
}
```

Note that in this example where no ids are requested, the list of not found ids will always be empty.

### 5.2.3. Explanation of response to /incoming

POST json key	json value (strings quoted)	
incoming	json list of json documents	list of incoming sms messages retrieved
notfound	json list of string	list of message ids for which no incoming message could be retrieved
from	string	sender phone number
to	string	recipient number (long number or short code)
id	string	id of the incoming SMS
message	string	the incoming message text
conversation	string	conversation id associated with this message (two-way SMS response)
resptoid	string	id of original SMS associated with this message (two-way SMS response)
origmess	string	original SMS text associated with this message (two-way SMS response, retrieved if requested)
time	string containing long integer	timestamp in milliseconds after the epoch (1970-01-01 00:00:00 Z)

### 5.2.4. Example Python 3 code

Example code for /incoming will be very similar to that for /status above

## Chapter 6. The `/incoming/single` endpoint

Used to retrieve one (1) incoming SMS that was either sent to a short or long number rented by the user, or to a number pool number in response to a two-way SMS.

Reading incoming SMS will by default mark them as read, unless the `markasread` parameter is explicitly set to `false`

This endpoint can either retrieve the next unread incoming SMS message, or retrieve an incoming SMS whose id is given (independent on whether it was read before or not).

Reading the a message will by default mark the message as read, but this can be avoided by setting the `markasread` parameter to `false`. This default value is `true` both when reading unread messages and when reading id listed messages. If `markasread` is set to `false`, the same message will in most cases be retrieved again on the next call.

### 6.1. POST request example

Probably from an application

```
https://secure.lekab.com/restsms/lekabrest/incoming/single
```

With the contents of the HTTP body:

```
{
  "username" : "testuser",
  "password" : "testpass",
  "markasread" : false,
  "getoriginal" : true
}
```

#### 6.1.1. Explanation of parameters for `/incoming/single`

The `/incoming/single` endpoint accepts a single string field as the "id" parameter, and if no id is given a maximum of one (1) unread incoming SMS is returned.

POST json key	json value (strings quoted)	
username	string	username of the LEKAB SMS sending account
password	string	password of the LEKAB SMS sending account
apikey	string	API key of the LEKAB SMS sending account
id	string	id of the SMS for which send status is to be retrieved
markasread	boolean (default true)	flag whether the incoming message should be marked as read (defaults to true)
getoriginal	boolean (default false)	should original SMS text in a two-way conversation be retrieved?

A POST call with default parameters, where the authorization credentials are supplied via Basic Authentication or X-Lekab-headers can supply either a zero-length body or an empty JSON document with the same result.

### 6.1.2. HTTP response to /incoming/single

A successful request (at least incoming SMS was retrieved) will return 200 OK and a json document of the following format. The Content-Type header of the response is application/json for all responses.

```
{
  "from" : "46701234567",
  "to" : "54321",
  "id" : "1077",
  "message" : "Please send more info about the club",
  "conversation" : "",
  "resptoid" : "",
  "origmess" : "",
  "time" : "1478538376000"
}
```

If the id given does not correspond to an incoming SMS that the requesting user is allowed to see, or if no id is given and no unread incoming SMS is available, a 404 Not found error will be returned.

### 6.1.3. Explanation of response to /incoming/single

POST json key	json value (strings quoted)	
from	string	sender phone number
to	string	recipient number (long number or short code)
id	string	id of the incoming SMS
message	string	the incoming message text
conversation	string	conversation id associated with this message (two-way SMS response)
resptoid	string	id of original SMS associated with this message (two-way SMS response)
origmess	string	original SMS text associated with this message (two-way SMS response, retrieved only if requested)
time	string containing long integer	timestamp in milliseconds after the epoch (1970-01-01 00:00:00 Z)